 wait(),notify(),notifyAll()不属于Thread类,而是属于Object基础类,也就是说每个对像都有wait(),notify(),notifyAll() 的功能.因为都个对像都有锁,锁是每个对像的基础,当然操作锁的方法也是最基础了. 先看java doc怎么说: wait导致当前的线程等待，直到其他线程调用此对象的 notify() 方法或 notifyAll() 方法。当前的线程必须拥有此对象监视器。该线程发布对此监视器的所有权并等待，直到其他线程通过调用 notify 方法，或 notifyAll 方法通知在此对象的监视器上等待的线程醒来。然后该线程将等到重新获得对监视器的所有权后才能继续执行. notify唤醒在此对象监视器上等待的单个线程。如果所有线程都在此对象上等待，则会选择唤醒其中一个线程。直到当前的线程放弃此对象上的锁定，才能继续执行被唤醒的线程。此方法只应由作为此对象监视器的所有者的线程来调用. "当前的线程必须拥有此对象监视器"与"此方法只应由作为此对象监视器的所有者的线程来调用"说明wait方法与notify方法必须在同步块内执行,即synchronized(obj之内). 调用对像wait方法后,当前线程释放对像锁,进入等待状态.直到其他线程(也只能是其他线程)通过notify 方法，或 notifyAll.该线程重新获得对像锁. 继续执行,记得线程必须重新获得对像锁才能继续执行.因为synchronized代码块内没有锁是寸步不能走的.看一个很经典的例子: package ProductAndConsume. import java.util.List. public class Consume implements Runnable{ private List container = null.

private int count. public Consume(List lst){ this.container = lst. } public void run() { while(true){ synchronized (container) { if(container.size()== 0){ try { container.wait().//放弃锁 } catch (InterruptedException e) { e.printStackTrace(). } } 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com